IEEE 2005 CUSTOM INTEGRATED CIRCUITS CONFERENCE

A Continuous-Time Programmable Digital FIR Filter

Y. W. Li, K. L. Shepard, and Y. P. Tsividis
Columbia Integrated Systems Laboratory
Columbia University, New York, New York 10027
Email: {ywli,shepard,tsividis}@cisl.columbia.edu

Abstract

We describe the design and implementation of a continuous-time finite-impulse-response processor chain, which includes a six-bit asynchronous ADC, an asynchronous digital core and an eight-bit asynchronous DAC designed in TSMC 0.25 μm technology. Continuous-time operation avoids aliasing and the quantization noise associated with the aliasing of harmonic components. We present measurement results demonstrating an audio low-pass-filter with a bandwidth of 6.0 kHz implemented with this processor.

Introduction

Conventional digital signal processing systems, which are discrete in time and discrete in amplitude (DTDA), are characterized by the deleterious effects of aliasing and quantization noise. Conventional analog systems, which process the signal continuously in time and amplitude (CTCA), do not suffer from these concerns. Analog systems, however, have high sensitivity to component tolerances and matchings, their dynamic range is comparatively low, and reconfigurability is limited and, usually, difficult. The merits and shortcomings of these systems are complementary, and this motivates a desire to find a signal processing system which combines the best attributes of both DTDA and CTCA systems. The focus of this work is to explore the relatively new area of systems which achieve exactly this combination of attributes by being discrete in amplitude but continuous in time (CTDA). Such systems promise to eliminate aliasing and significantly reduce in-band quantization noise while still providing the high dynamic range, robustness, and reconfigurability of DTDA systems.

CTDA systems [1] are time-encoded, that is, they use the temporal positioning of samples to carry information [2] [3]. Since CTDA systems are "clockless," asynchronous design techniques are a natural choice. However, it is important to note that conventional asynchronous design techniques are employed in the design of DTDA system in which only the relative "ordering" of samples is preserved [4]; an implicit sample time is assumed based on the clock frequency of the analog-to-digital (ADC) converter. In contrast, this paper describes the design of an asynchronous digital processor that preserves the *time interval between samples* since in a CTDA representation, this carries information.

Benefits of CTDA Systems

When a signal is sampled at a sampling frequency $\pm f_s$, the signal is converted into a digital sequence with implicit timing information. Aliasing results as there is no way to distinguish the frequency component ($\pm f_{signal}$) from the aliased components ($p \cdot f_s \pm f_{signal}$) where p is an integer. The elimination of sampling in CTDA systems results in the elimination of these aliasing effects.

[1] presents a perspective to understand how quantization noise is reduced in CTDA systems. Quantization noise in conventional digital signal processing systems results from the aliasing of the harmonic components resulting from quantization. Hence, if sampling is eliminating, no harmonics will be aliased to the in-band frequency spectrum, a significant advantage for continuous-time digital signal processing.

This work was supported in part by the NSF under Grant CCR-00-86007.

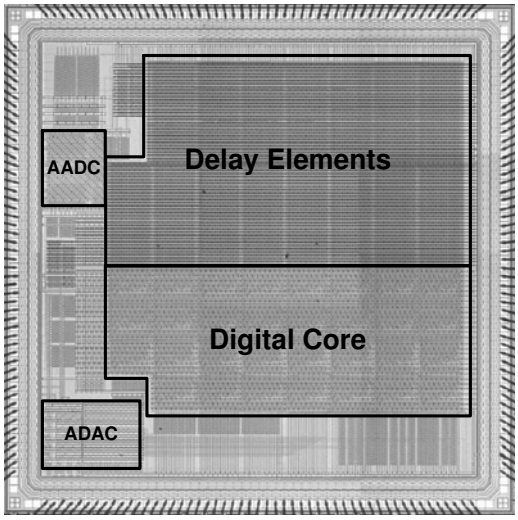

Fig. 1. Die photo of the CTDA asynchronous filter in a TSMC $0.25\mu m$ process

Overall chip architecture

In this paper, we describe the detailed design of a complete CTDA FIR filter fabricated in a TSMC 0.25 μm CMOS technology, the die photo of which is shown in Fig. 1. The 25 mm^2 continuous-time digital FIR filter consists of a six-bit analog-to-digital converter (ADC), an asynchronous 16-tap digital FIR filter, and a digital-to-analog converter (DAC) for analog waveform reconstruction, all operating asynchronously. The ADC functions as a delta modulator, outputting a single bit representing whether the new sample is one quantization level higher or lower than the preceding sample. Accumulation is done in the digital filter to 16 bits of precision. The result from the digital core is passed to an eight-bit current-steering DAC. In the chip, a bundled-data approach is employed throughout in which a request (REQ) signal accompanies the data, allowing for data-independent matched timing through the dataflow elements. Unlike discrete-time asynchronous systems, there is no acknowledgment handshake because no backpressure can be exerted to data movement. Because time intervals must be preserved, downstream processing elements must be able to immediately process data tokens.

Asynchronous analog-to-digital converter (AADC). Analog-to-digital conversion proceeds in a continuous-time manner in that samples are generated when the input analog signal crosses predetermined quantization levels [5]. The delta-modulation architecture, shown in Fig. 2, considerably simplifies the design of the delay elements and digital FIR hardware. The six-bit resistor-string DAC generates two references voltages, V_{upper} and V_{lower}, separated by an LSB which "enclose" the current input voltage value. If the input voltage level crosses V_{upper} (V_{lower}), INC (DEC) goes to logic-1, with DEC (INC) still zero. In response, the controller generates a REQ pulse with UP a logic-1 (logic-0). The six-bit $current\_value$ is incremented (decremented) by one and decoded to the one-hot dac_decode to

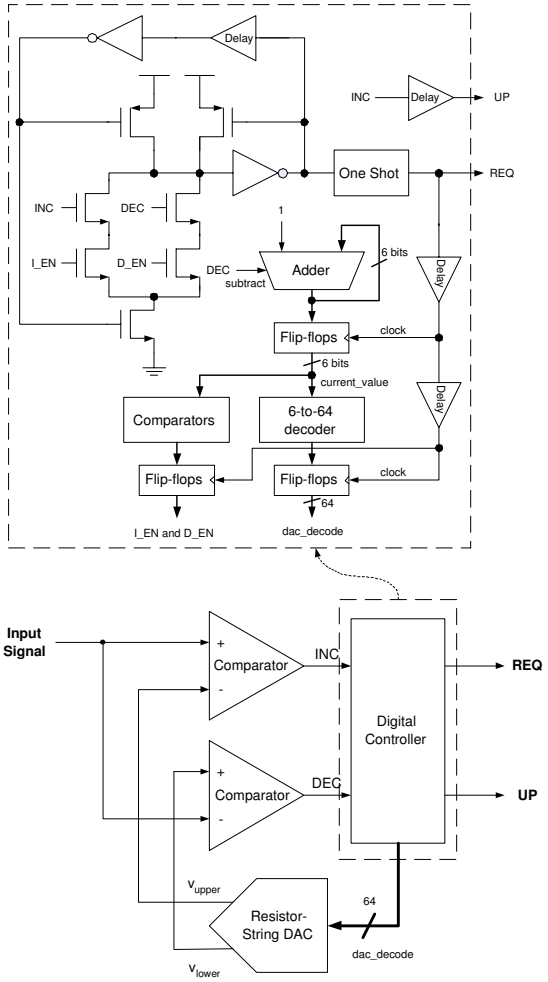

Fig. 2. Asynchronous ADC block diagram

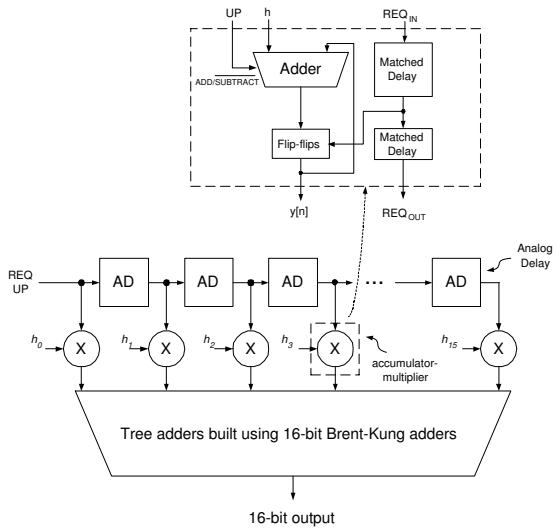

Fig. 3. Asynchronous FIR filter block diagram

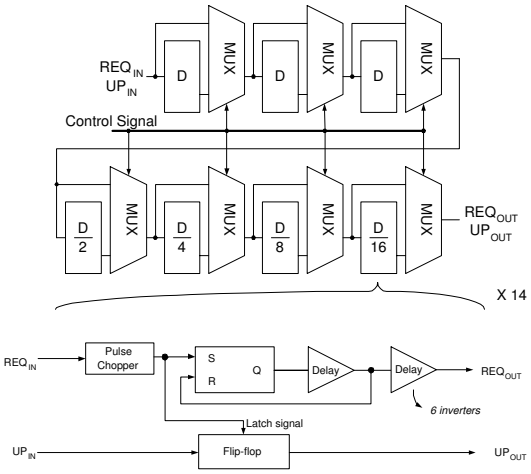

Fig. 4. Digitally tunable analog delay line

control the DAC.

We define T_{loop} as the loop delay of the ADC; that is, the combined delay of the controller, DAC, and comparator. The I_EN and D_EN signals are deasserted when the input signal goes out of range to prevent new data from being sent from the ADC. The self-resetting initial stage has a reset delay (T_{reset}) that ensures that once this stage has evaluated (by the action of the INC or DEC signals), it cannot reevaluate again for a time T_{reset}. Setting $T_{reset} > T_{loop}$ ensures that spurious requests are not generated while the ADC is settling. In our design $T_{reset} \cong 30\ nsec$, while $T_{loop} \cong 20\ nsec$. The one-shot circuit generates the REQ pulse, which is approximately 500 psec in width. This pulse is also used to clock the internal state latches of the controller.

The bandwidth of the input signal determines the maximum allowable T_{loop} of the ADC. A sine wave of frequency f and amplitude A has peak derivative of $2\pi Af$. For n-bit data conversion, the maximum allowable value for T_{loop} is determined by the minimum time (T_{sample}) for the input signal to traverse one LSB, $T_{sample} = 1/\left(2\pi f 2^{n-1}\right)$. In the audio applications considered here, we assume that the input signal is bandlimited to 22 kHz and that a 22 kHz sine wave is input to the AADC. In this case, for six-bit conversion, $T_{loop} < T_{sample} = 0.226\ \mu sec$.

The comparators [6] provide a gain of between 72 dB and 102 dB over the common-mode input voltage range from 1 to 2 V, providing a comparator resolution ($V_{resolve}$) of better than 0.3 mV. For slowly changing inputs ($f < \frac{1}{2\pi A}\frac{V_{resolve}}{T_{loop}}$, where A is the amplitude of the input signal), it may be possible for the INC or DEC signal to trigger the dynamic pull-down stage at the input of the controller (Fig. 2) with non-full-rail inputs, resulting in a variation in the delay to the rising edge of REQ. The only deleterious effect of this nonlinearity is some additional harmonic distortion in the AADC output that is usually out-of-band.

Asynchronous finite-impulse-response filter. The basic dataflow structure of the digital filter is shown in Fig. 3, implementing the FIR function:

$$y(t) = \sum_{k=0}^{15} h_k x(t - kT_{delay})$$

where T_{delay} is the delay of one of the AD (analog delay) blocks of Fig. 3 and the h_k are the 8-bit filter coefficients.

The design of the AD blocks presents considerable challenges. Given the low-bandwidth of the input signals, this delay block must present a nominal delay time of approximately $T_{delay} = 12.5\ \mu sec$, which is quite large compared to the fanout-of-four (FO4) delay of this technology ($\cong 100\ psec$). Furthermore, this AD block must have sufficient granularity, that is, have enough delay elements, to ensure that it has the capacity to store the requisite number of REQ pulses. In particularly, for six-bit

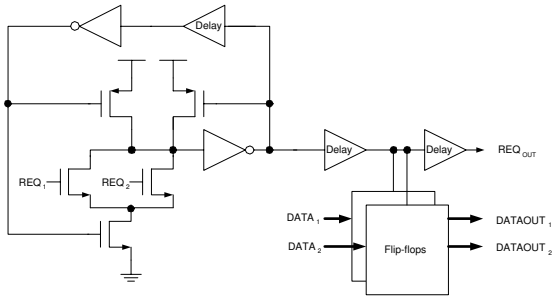

Fig. 5. Join control used in the adder tree

operation, there could be as many as $T_{delay}/T_{sample} = 56$ request pulses in the line, while for 8-bit operation, there could be as many as 224 request pulses in the line. Building fewer stages with longer buffer delays may not result in a fast enough response time to distinguish two pulses coming at the closest separation. In order to minimize the chip area, we need to optimize the ratio $\frac{delay}{area}$ for a buffer while keeping the delay low enough to distinguish two back-to-back requests.

The detailed implementation of one of the AD blocks is shown in Fig. 4 and includes 10,584 inverters per delay block. We design the delay line to accommodate an eight-bit ADC, even though only a six-bit ADC is used in the current implementation. We subsequently scaled back the resolution to six bits to ease the design requirements on T_{loop} but kept the delay line with the eight-bit specifications. As a result, the delay line is "longer" and consumes more area than would have been required for a six-bit implementation. The AD blocks control the delay of the REQ signals, to which the one-bit output of the ADC is synchronized. Because an inverter will inevitably have a small beta ratio skew, either by unintended design or process variations, a pulse when passed through a large number of inverters will, in general, be runted or filtered out entirely. As a result, regeneration is required and is done after every six inverters by means of the SR-latch and one-shot as shown in Fig. 4. Finally, the delay is digitally tunable with $D = 6.4\,\mu s$, allowing any delay between $0.4\,\mu s$ and $25.2\,\mu s$ with a resolution of $0.4\,\mu s$ to be chosen.

The accumulator-multipliers are implemented without hardware multipliers as shown in Fig. 3. A pipelined Brent-Kung adder tree is used to add the sixteen partial sums of the accumulator-multiplier blocks. The pipeline stage delay (T_{pipe}) must be less than the shortest interval between any two samples to avoid the collision of data "waves" within a single pipeline stage.

Datapath joins in the Brent-Kung tree introduce unique challenges to the design. Consider the case of two samples which are separated by exactly T_{delay}, the delay of a single analog delay (AD) element. In this situation, the two requests will arrive at the inputs of the adder at exactly the same instance. By this example, it is clear that the inputs to any of the blocks in the Brent-Kung tree can arrive arbitrarily close together. In this case, a mechanism must exist to allow one of the requests to be processed and to discard the other. In particular, this will happen if the two requests are closer together than $T_{separation}$, where $T_{separation} > T_{pipe}$. Fig. 5 is the "join control" circuit used for this purpose. $T_{separation}$ is defined by the self-resetting loop delay. Infrequent metastabilities in the flip-flops are possible in the case of "collisions" resolved by the join controls. These result in harmonic distortion that is likely to be well outside the signal bandwidth of interest and can be easily removed by a reconstruction (or smoothing) filter if not already removed by the finite bandwidth of the DAC. In our design, $T_{separation} = 6\ nsec$ and $T_{pipe} = 1.5\ nsec$.

Asynchronous digital-to-analog converter (ADAC). The eight most significant bits of the asynchronous adder[1] are presented to the DAC, implemented with a current-steering architecture as shown in Fig. 6, to produce the analog output. The digital controller of the DAC is similar to the one in the ADC, employing a self-resetting initial stage that ensures that the DAC output has stabilized before a new request is processed. The worst-case settle time of the DAC, which determines the minimum possible time between samples, is approximately 40 nsec. The decoder converts the two's complement output from the digital core into the decoded representation required for the current-source array.

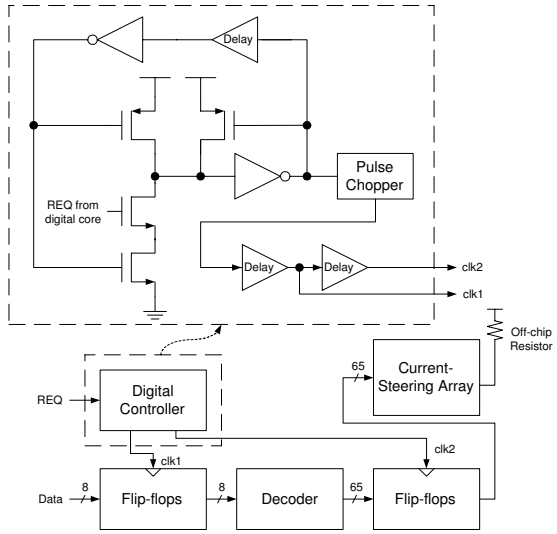

Fig. 6. Asynchronous DAC block diagram

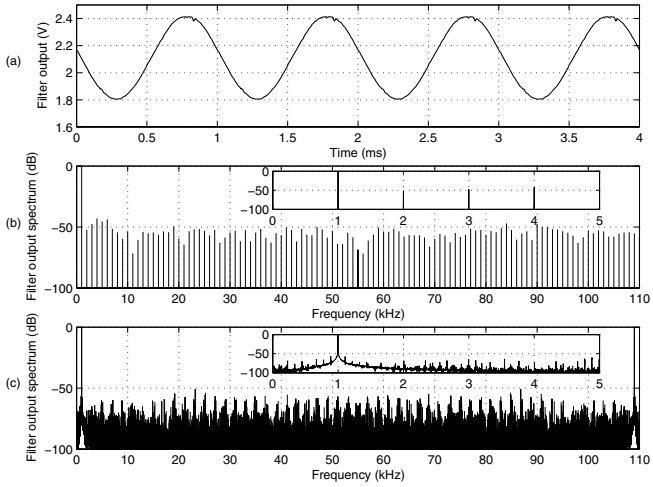

Fig. 7. (a) Measured output of the ADAC for a 1-kHz sinusoidal input to the AADC with the filter configured as an all-pass; (b) corresponding frequency spectrum of the ADAC output (inset: expanded view up to 5 kHz); (c) simulated frequency spectrum of an ideal six-bit converter operating at $f_s = 110kHz$ (inset: expanded view up to 5 KHz).

Hardware Measurement Results

Measured properties of the CTDA filter are summarized in Table I.

A. Asynchronous ADC and DAC performance

To measure the performance of the data conversion circuits and elucidate the unique features of the continuous-time imple-

[1] A programmable shifter exists at each stage of the adder tree to normalize the result and prevent overflow.

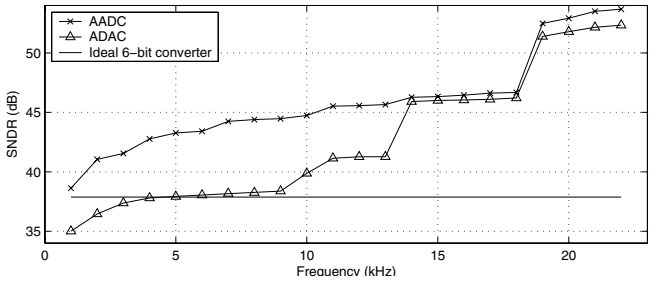

Fig. 8. Signal-to-noise-and-distortion ratio (SNDR) at the output of the AADC and ADAC as a function of input frequency.

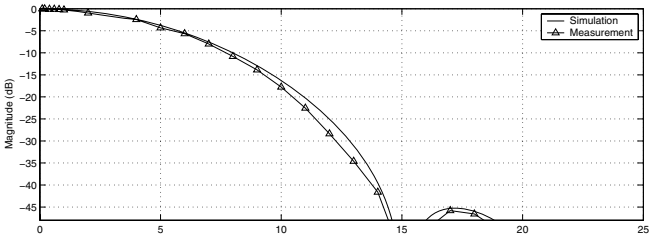

Fig. 9. Low-pass filter transfer characteristic.

mentation, we configure the filter as an all-pass and measure both the digital output of the ADC (accumulated from the UP signal) and the analog output of the DAC.

Fig. 7(a) shows the *measured* output of the DAC for a 0-dbFS, 1 kHz sinusoidal input to the ADC. Fig. 7(b) is the corresponding frequency spectrum, computed from an FFT with a Welch window function to a frequency resolution of 25 Hz. The inset of Fig. 7 shows the features of the spectrum in the range up to 5 kHz. Only the discrete harmonics of the input frequency are evident. For these measurements, the analog waveform is captured and quantized to a resolution of $\Delta t = 4\ ns$ by a sampling oscilloscope with the "noise floor" determined by this quantization below -140 dB. For contrast, the simulated spectrum of a reconstructed "ideal" six-bit ADC with a sampling rate of $5 \cdot 22$ kHz (2.5 times the Nyquist rate) is shown in Fig. 7(c). In this case the aliased component of the input tone is evident at 109 kHz as is aliased harmonic distortion which appears as in-band quantization noise.

Fig. 8 shows the measured signal-to-noise-plus-distortion ratio (SNDR) (for a 55-kHz band) at both the output of the AADC and the output of the ADAC. SNDR is shown as a function of input signal frequency for a full-scale sinusoidal input signal. The "classical" quantization noise value, as given by $6.02n+1.76\ dB$ for an n-bit converter, is shown for six bits (37.8 dB). The SNDR improves for input tones higher in the target bandwidth, since fewer in-band harmonics are supported. The "steps" around 18 kHz occur as the third harmonic of the input frequency are pushed out of band. Similar steps around 13 kHz occur as the fourth harmonic is pushed out of band. This harmonic is clearly more pronounced at the ADAC output, due to additional harmonic distortion introduced there. At 22 kHz, the SNDR of the AADC and ADAC are 53.68 and 52.34 dB respectively, which is better than the quantization noise floor for an ideal eight-bit converter.

B. Overall filter characteristics

To characterize the entire filter, the filter is programmed to be low-pass with the cutoff at 6.0 kHz; the coefficients are obtained using the window method with a Hamming window. Fig. 9 shows the magnitude of the filter transfer function as measured from 100 Hz to 25 kHz. At 14.5 kHz, the attenuation is more than -48 dB, reaching the resolution limit of the 8-bit ADAC. The measured results are compared in Fig. 9 with the simulated ideal filter response.

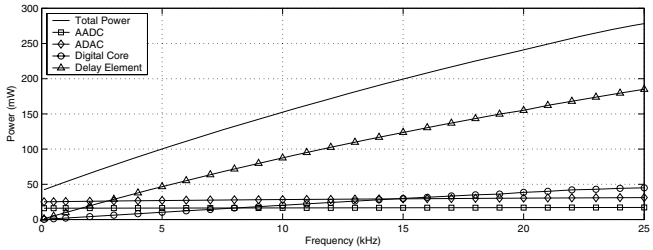

Fig. 10. Power measurements on components of filter as a function of input frequency.

TABLE I
SUMMARY OF MEASURED SYSTEM CHARACTERISTICS

System	Area	25 mm^2
	Power: 1 to 22 kHz (0 dBFS tone)	42.02 mW
	SNDR (over 55 kHz bandwidth)	35.01 to 52.34 dB
AADC	Area	0.27 mm^2
	SNDR (over 55 kHz bandwidth)	38.62 to 53.68 dB
	Number of bits	6
	Power: 1 to 22 kHz / -0.1 dBFS tone	16.08 to 16.97 mW
ADAC	Area	0.42 mm^2
	Number of bits	8
	Power: 1 to 22 kHz / -0.1 dBFS tone	25.50 to 30.80 mW
Delay Elements	Area	6.60 mm^2
	Power: 1 kHz / 0 dBFS tone	9.84 mW
Digital Core	Area	5.06 mm^2
	Power: 1 to 22 kHz (0 dBFS tone)	2.00 mW to 42.02 mW

C. Power measurement

In the conventional digital signal processor, we have to sample the signal at a rate at least twice the bandwidth of the signal (Nyquist rate) for reconstruction. The power consumption is a function of the sample rate but independent of the actual spectral content of the input signal. In contrast, the power consumption of the CTDA filter implemented here is a strong function of the spectral content of the input signal. Fig. 10 shows the measured power consumed by various components of the filter as a function of the frequency of a full-scale input sinusoid. The delay elements and digital core show a close to linear dependence of power on input frequency. For the AADC and ADAC, which are dominated by static bias currents, the power is virtually independent of frequency. The power is dominated by the delay elements, which is designed to support eight-bit resolution; six-bit resolution would require only one-fourth the number of delay stages, allowing a factor-of-four power reduction.

Conclusion

We have demonstrated the design of a continuous-time, discrete-amplitude digital signal processor. Such a design leverages the noise immunity and robustness of digital systems with non-sampled operation to avoid aliasing and reduce quantization noise.

References

[1] Y. P. Tsividis, IEEE Int. Conf. Acoustics, Speech, and Signal Processing, vol. II, pp. 589-592, 2004.
[2] H. Inoue et al., Electronics Letters, vol. 2, pp. 95-96, 1966.
[3] J. Foster et al., IEEE SoutheastCon, vol. 2, pp. 861-863, 1991.
[4] Y. W. Li et al., IEEE JSSC, pp. 704-708, April 2004.
[5] E. Allier et al., IEEE Int. Symposium on Async. Circuits and Systems, pp. 196-205, 2003.
[6] M. Bazes, IEEE JSSC, pp. 165-168, February 1991.